

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-295871

(43) 公開日 平成7年(1995)11月10日

(51) Int.Cl.*

G 0 6 F 12/00

17/30

識別記号

5 1 8 A

5 3 3 J

庁内整理番号

7608-5B

7608-5B

9194-5L

F I

G 0 6 F 15/ 40

3 1 0 C

技術表示箇所

審査請求 未請求 請求項の数11 O L (全 21 頁)

(21) 出願番号 特願平7-23143

(22) 出願日 平成7年(1995)2月10日

(31) 優先権主張番号 2 2 8 3 1 9

(32) 優先日 1994年4月15日

(33) 優先権主張国 米国 (US)

(71) 出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSINESS MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州アーモンク (番地なし)

(72) 発明者 アラン・シー・アザギュリ

イスラエル20306ネシャ、ハシクマ・ストリート 32/6

(74) 代理人 弁理士 合田 潔 (外2名)

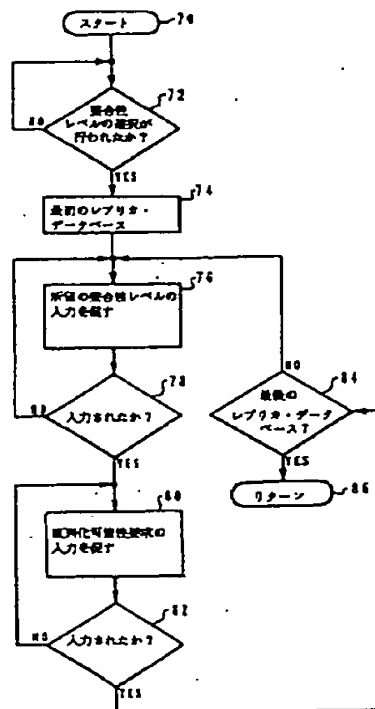
最終頁に続く

(54) 【発明の名称】 データベース・アクセス効率の向上方法及びシステム

(57) 【要約】

【目的】 分散データ処理システム内でのデータベース・アクセス効率を向上させる方法とシステムを提供する。

【構成】 分散データ処理システム内の1つまたは複数のロケーションで実施される回復可能なデータベース・システムにおいて、プライマリ・データベースとレプリカ・データベースとの間に維持される複数の多様な整合性レベルが指定される。ユーザは各レプリカに適した整合性レベルを選択できる。プライマリ・データベース内のレコードに対するアップデートがあると、各レプリカ内の対応するレコードに対するアップデートは、各レプリカに対して選択された整合性レベルに一致する方法で行われる。これにより、プライマリ・データベースと完全に整合性のあるレプリカが提供できる。あるいは、完全に整合性のあるデータを必要としないアプリケーションには、第2のレプリカを提供し、アクセス効率を向上させることができる。



【特許請求の範囲】

【請求項1】 分散データ処理システム内でのデータベース・アクセス効率を向上させる方法であって、前記分散データ処理システム内において、ストアされた複数のレコードを持つ1つのデータベースを選択しプライマリ・データベースとして指定するステップと、前記分散データ処理システム内の第2の物理的ロケーションにおいて前記プライマリ・データベースを複製するステップと、

前記プライマリ・データベースと前記の複製されたレプリカ・データベースとの間に維持される複数の多様な整合性レベルを指定するステップと、

ユーザに前記の複数の多様な整合性レベルから1つを指定させるステップと、

を有し、

前記プライマリ・データベース内のレコードに対するアップデートに回答して、前記プライマリ・データベースと前記レプリカ・データベースとの間に前記の複数の多様な整合性レベルの前記の指定された1つを自動的に維持する前記方法。

【請求項2】 前記分散データ処理システム内の第3の物理的ロケーションにおいて前記プライマリ・データベースを複製するステップをさらに有する、請求項1に記載の方法。

【請求項3】 ユーザに前記の複数の多様な整合性レベルから1つを指定させる前記ステップが、前記プライマリ・データベースとそのそれぞれのレプリカとの間に維持される前記の複数の多様な整合性レベルから別の1つを指定させるステップを有する、請求項2に記載の方法。

【請求項4】 分散データ処理システム内でのデータベース・アクセス効率を向上させるシステムであって、前記分散データ処理システム内において、ストアされた複数のレコードを持つプライマリ・データベースと、前記分散データ処理システム内の第2の物理的ロケーションにおいてストアされている前記プライマリ・データベースのレプリカと、

前記プライマリ・データベースと前記レプリカ・データベースとの間に維持される複数の多様な整合性レベルを指定する手段と、

ユーザに前記の複数の多様な整合性レベルから1つを指定させる手段と、

前記プライマリ・データベース内のレコードに対するアップデートに回答して、前記プライマリ・データベースと前記レプリカ・データベースとの間に前記の複数の多様な整合性レベルの前記の指定された1つを自動的に維持する手段と、

を有するシステム。

【請求項5】 前記分散データ処理システム内の第3の物理的ロケーションにおいてストアされている前記プラ

イマリ・データベースの第2のレプリカをさらに有する、請求項4に記載のシステム。

【請求項6】 ユーザに前記の複数の多様な整合性レベルから1つを指定させる前記手段が、前記プライマリ・データベースとそのそれぞれのレプリカとの間に維持される前記の複数の多様な整合性レベルから別の1つを指定させる手段を有する、請求項5に記載のシステム。

【請求項7】 分散データ処理システム内でのデータベース・アクセス効率を向上させる方法であって、

10 前記分散データ処理システム内において、ストアされた複数のレコードを持つ1つのデータベースを選択しプライマリ・データベースとして指定するステップと、

前記分散データ処理システム内の第2の物理的ロケーションにおいて前記プライマリ・データベースを複製するステップと、

前記プライマリ・データベース内の1つの選択されたレコードに対するアップデートに回答して、前記の複製されたデータベース内の対応する選択されたレコードへのアクセスを自動的にロックするステップと、

20 前記の複製されたデータベース内の前記の選択されたレコードをアップデートするステップと、

前記アップデートの後で、前記の選択されたレコードへのアクセスをリリースするステップと、

前記分散データ処理システム内のユーザに、前記プライマリ・データベースおよび前記の複製されたデータベースの中のレコードを照会させるステップと、

を有し、データの絶対的な整合性を維持するとともにデータアクセスの使用可能性を向上させる方法。

【請求項8】 前記プライマリ・データベース内のレコードに対する各アップデートが、コミット・オペレーションを行う前の1つの整合性時点に戻り、前記の選択されたレコードへのアクセスを前記アップデートの後でリリースする前記ステップが、前記アップデートに続くコミット・オペレーションの後でのみ前記の選択されたレコードへのアクセスをリリースするステップを有する、請求項7に記載の方法。

【請求項9】 分散データ処理システム内でのデータベース・アクセス効率を向上させるシステムであって、前記分散データ処理システム内において、ストアされた

40 複数のレコードを持つプライマリ・データベースと、

前記分散データ処理システム内の第2の物理的ロケーションにおいてストアされている前記プライマリ・データベースのレプリカと、

前記プライマリ・データベース内の1つの選択されたレコードに対するアップデートに回答して、前記の複製されたデータベース内の対応する選択されたレコードへのアクセスを自動的にロックする手段と、

前記の複製されたデータベース内の前記の選択されたレコードをアップデートする手段と、

50 前記アップデートの後で、前記の選択されたレコードへ

のアクセスをリリースする手段と、前記分散データ処理システム内のユーザに、前記プライマリ・データベースおよび前記の複製されたデータベースの中のレコードを照会させる手段と、を有し、データの絶対的な整合性を維持するとともにデータアクセスの使用可能性を向上させるシステム。

【請求項10】前記プライマリ・データベースを、いかなる時点でも、コミット・オペレーションを行う前の整合性時点に戻す手段をさらに有する、請求項9に記載のシステム。

【請求項11】前記アップデートの後で前記の選択されたレコードへのアクセスをリリースする前記手段が、前記アップデートに続くコミット・オペレーションの後でのみ前記の選択されたレコードへのアクセスをリリースする手段を有する、請求項10に記載のシステム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、改善されたデータベース・アクセス制御の方法に関し、具体的には、プライマリ・データベースと1つまたは複数のレプリカ・データベースとの間の整合性を保つ改善された方法とシステムに関する。より具体的には、本発明は、回復力のある(resilient)データベース・システムにおいて、ユーザが、複数のレプリカ・データベースに付ける多様な整合性レベルを選択出来る、改善された方法とシステムに関する。

【0002】

【従来の技術】多目的のマルチプロセッシング・コンピュータ・システムは、典型的には、通信網によって相互接続された複数のノードを含む。そのようなシステムでは、各ノードは、データ処理装置、データ記憶装置、および、複数の通信ポートを含むことができる。データ処理装置は、複数のオペレーティングシステムの要素の制御の下でのマルチプログラミング・モードで実行している場合があり、この場合、データ処理装置は、複数のノードとみなされる。典型的には、データ記憶装置は、データファイル、オペレーティング・システムとその情報管理要素、および、ユーザのアプリケーション・プログラムをストアしている。

【0003】データは、企業にとっての重要な側面からビジネスを抽出した情報である。そのようなシステムでのチャレンジは、ビジネスニーズに合った使用可能性、パフォーマンスおよび費用でエンドユーザがデータにアクセスできる方法で、システムのデータ記憶と通信資源を使用することである。データへのアクセスは、データの整合性と完全性を確実にするために制御されなければならない。分散データ処理システム環境でのデータアクセスの付加的な特徴は、地理的および時間的親和性である。

【0004】分散されたデータの構造の基礎は地理的親

和性である。或るデータ項目へのアクセスは、地理的にまとまる傾向がある。データのダイナミックな複製方法の基礎は時間的親和性である。最近アクセスされたデータ項目は、最近アクセスされなかったデータ項目よりも、近い将来においてアクセスされる可能性が高い。或るデータ項目に対するアクセスがまとまる傾向があるノードは、親和性ノードと呼ばれる。或るデータ項目に対する親和性ノードは前もってわからず、また、そのノードは時間によって変化する。

- 10 【0005】分散データ技術は、データのある場所(データロケーション)、データ共有(データシェアリング)の度合い、通信網全体にわたって提供されるデータベース制御の度合い、および、データアクセスの型の属性にしたがって分類することができる。データロケーションは、集中管理、分割化、または、複製化に分けられる。データシェアリングの度合いは、集中化、非集中化、または、分散化に分けられる。データベース管理の制御は、ユーザ提供のもの(分散データ)、または、システム提供のもの(分散データベース)に分けられる。
- 20 データアクセスは、トランザクション処理型、機能処理型、または、データ処理型に分けられる。

【0006】歴史的には、集中管理型がデータベース記憶とアクセスを管理するために使用されてきた。そのような処理手法においては、データ管理とアプリケーション処理が集中管理される。単一のデータベース・マネージャが使用され、ユーザを中央システムに接続するためにテレプロセッシング通信網が使用される。集中処理手法の変形として、処理の或るものが、通信網の中のノードに分散される。しかし、データは集中管理される。

- 30 【0007】データベースの集中管理手法の利点は、以下の通りである。(1)データベースの整合性が単一のデータベース・マネージャによって確実にできること、(2)すべてのアプリケーション・プログラムが単一のアプリケーション・プログラミング・インターフェースにしたがって作成することができ、すべてのデータが1つのロケーションにストアされているので、アプリケーション・プログラムはデータロケーションを承知していなくてもよいこと、(3)集中データ処理環境でのデータ管理の問題を解決できる多くの有効なツールがあること、(4)単一システムは、操作し、維持し、制御するのがより容易であること。

【0008】しかし、データベースの集中処理手法には、いくつかの不利な点がある。すなわち、(1)ある企業にとっては通信コストが高くつき、アプリケーション・プログラムの性能が通信遅延のためにさがること、

(2)データ使用可能性が通信網あるいは中央システムの不安定性によって悪くなることがあり、これらの問題はバックアップ・システムや通信網に冗長度を持たせることによって解決されねばならないこと、(3)ある企業にとっては、単一の中央処理システムの処理能力がす

でに限界に達していること。

【0009】分散データ処理システムのノードにデータを分散するには、一般的に2つの手法がある。これらの手法とは、分割化と静的複製化である。分割データ手法にはデータベースのプライマリ・コピーがないが、複製化手法にはデータベースのプライマリ・コピーがある。

【0010】分割データベース手法は、データベースを明確な区画に分割し、これらが、ノードに分散される。したがって、あるデータ項目は、1つのノードにだけ存在する。各ロケーションは、そのロケーションでデータを管理するデータベース・マネージャーを持つ。データ分散マネージャーはアプリケーション・プログラムのデータ・リクエストを受け、データがそのロケーションすなわちローカルに存在する場合はそのリクエストをローカルリクエストにマップし、データが別のロケーションにある場合は、そのリクエストをリモートリクエストにマップする。

【0011】分割された分散データベースでは、リクエストされたデータがローカルに存在する場合には、良いデータ使用可能性とアクセス性能が得られる。さらに、各データ項目が単一のデータベース・マネージャーによって管理されるので、データベースの整合性が容易に得られる。よい分割アルゴリズムが前もってわかり、それが存在し、且つ、安定したものであるならば、これらの結果を得ることができる。

【0012】上述した分割されたデータベースでは、複数のロケーションでデータを変更するプログラムのために、システムが、通信網全体にわたる回復処理を提供しなければならない。

【0013】分割されたデータベース・システムには、また、以下の不利な点がある。(1)分割アルゴリズムがデータアクセス・パターンに一致しない場合には、データの使用可能性と性能が落ちること、(2)アプリケーション・プログラムが、データロケーション、あるいは、少くともデータ分割アルゴリズムを承知していなければならない、また、データロケーションによってデータベースを異なった方法でアクセスしなければならないこと、(3)データベース分割アルゴリズムを変更することは、データロケーションが各ノードのアプリケーション・プログラムの中、エクシット、または、デクラレーションに反映されているので、非常に難しいこと、

(4)各ノードにおける既存データの再配置とアルゴリズムの変更は通信網全体にわたって同期して行われねばならず、したがって、最適な性能とデータの使用可能性を維持するのに必要な分割アルゴリズムを調整することができない場合があること、(5)分割されたデータベース全体にわたって或るデータ項目を均一にアクセスするか、または、全データベースをアクセスしなければならないプログラムは性能とデータ使用可能性の低下を蒙ること。

【0014】データを分散するための静的複製手法は、中央ノードを含むか、または、含まない手法を含む。前者の場合、中央ロケーションはデータベースのプライマリ・コピーをストアし、各ロケーションはデータベース・マネージャーとデータベースのコピーを持つ。静的複製手法の典型的な使用方法では、プライマリ・データベースがコピーされて各レプリカ・ロケーションすなわちノードに送られ、各ノードでのローカル処理のためにデータが使用できるようになる。各レプリカ・ロケーションで行われたデータ変更は、プライマリ・データベースに対して後で処理を行うために集められる。アプリケーション・プログラムが処理を行う間をぬって、ローカルで行われたデータ変更が中央ロケーションに送られ、プライマリ・データベースに対してアップデートが行われる。レプリカ・データベースを管理するこの手法は複数回のアップデートを防ぐ手法を何も持たないので、プライマリ・データベースに対する各アップデートを手作業で発見して解決するか、あるいは、そのような複数回のアップデートが起こらないようにアプリケーション・プログラムを何らかの方法で制限しなければならない。プライマリ・データベースがレプリカ・コピーのアップデートに一致するようにした後、新しいコピーがレプリカ・ロケーションに送られ、全プロセスが再び開始される。

【0015】プライマリ・コピーを中央ロケーションに持つ静的複製の主要な利点は、すべてのデータがローカルにアクセス可能であるので、高いデータ使用可能性と良い応答時間が得られる。しかし、この方法には以下の不利な点がある。(1)システムが複数回のアップデートを防ぐことができないのでデータベースの整合性を保つことが難しく、静的レプリカのために実行可能なデータベース処理が厳しく制限されること、(2)システムが、アプリケーション・プログラムのアクセスが必要とする最新データを保証できないこと、(3)オペレーション上の特別なプロシージャが、レプリカ・データの変更を集めプライマリ・データベースをアップデートするために必要になり、このプロシージャを実行することは高くつき、また、エラーを起こし易いこと、(4)データのアップデートの間にはデータ伝送のために不必要に大きいバンド幅を必要とし、確認のための十分大きなウィンドウを提供することは多くのアプリケーションでは実行不可能であり、レプリカがオペレーションの合間の狭いウィンドウの中だけで送られるので、確認プロシージャの間は、データベースが使用できなくなる可能性が大きいこと。さらに、1つあるいは複数のノードが動作不可能になった場合、予定されたウィンドウの中で確認プロシージャが行えなくなる。

【0016】上述した静的複製の基本的な手法に関して、多くのさまざまな手法が文献に記載されている。たとえば、複数回のアップデートが起こらないようにアプ

リケーション・プログラムを設計することができるし、あるいは、レプリカは読み取りアクセスだけに制限することもできる。あるいは、アプリケーション・プログラム自身が、後でプライマリ・ロケーションに送るためにアップデートを集め、プライマリ・ロケーションにおいて、データベース・マネージャのログからこの情報を探り出すようにすることもできる。レプリカ・ロケーションでは、全面的なレプリカまたは部分的なレプリカだけを形成することもできる。レプリカ・データベース全体あるいは、保持されているデータへの変更のみを送るようにすることができる。トランザクションによって行われた変更を種々のノードに送り、トランザクション終了処理の1部としてアクノレージメントを受領することによって、レプリカの同期性を常に保つようにすることができる。そのような同期手法によって静的複製の整合性の問題を解決することができる。しかし、これらの手法では、そのようなシステムの性能と使用可能性の利益の多くを失う。

【0017】米国特許4,007,450号には、各ノードが他のノードと或る種のデータセットを共有し、中央ロケーションには絶えず同期されるレプリカだけがあるがプライマリ・コピーがない、分散データ制御システムが記載されている。各ノードは、他のノードのいずれかがアップデートしようとしていないかぎり、共有されたどのデータセットをもアップデートするように作動し、複数のノードがアップデートしようとしているときには、高いプライオリティーを持つノードがアップデートを行う。各ノードはそのメモリに、共有された各データセットのノードロケーションと、共有された各データセットに対して各ノードが持つアップデート・プライオリティーをストアしている。或るノードでデータセットがアップデートされるとき、レプリカを持っているすべてのノードにそのアップデートが送られる。上記のような手法は、静的複製の整合性問題を解決することはできるが、性能と使用可能性の利益の多くが失われる。

【0018】米国特許4,432,057号には、分散マルチプロセス・データベース・システムにおける資源の活用を制御するための分散システム制御の下での、データのダイナミックな複製方法が記載されている。このシステムでは、指定された現在性 (currency) を持つデータへのアクセスリクエストが許され、アップデートされたデータの確認が、各ノードにおいて実行される制御プロセスを使用することによって選択的に行われる。この場合、各ノードは、他ノードにおける共有データ項目のステータスに関するそのノードの見方をあらし各ノードにファイルされているステータス・アンド・コントロール (status and control message:SAC) メッセージを使用する。このシステムでは、各データベース・リクエストはインターセプトされ、ローカル・コピーが現在性に対する要求を満たすかどうか判断される。現在性

が満たされない場合、そのリクエストを実行する前に、関連するノードと交渉を行うことが必要である。したがって、レプリカ・データベースの現在性は、アクセス・パターンに従ってダイナミックに変わる。

【0019】上記のことから、データ処理システムの1つまたは複数のロケーションにおいて複製されるジャーナルされたデータベースを含む回復力のあるデータベース・システムで使用し、それにより、各レプリカ・データベース内で維持される整合性レベルを、任意に且つ選択的に割り当てることができる方法とシステムの必要性があることがわかる。

【0020】

【発明が解決しようとする課題】本発明の目的は、分散データ処理システム内のデータベース・アクセス制御のための改善された方法を提供することである。

【0021】さらに、本発明の目的は、分散データ処理システム内のプライマリ・データベースと1つまたは複数のレプリカ・データベースとの間の整合性を維持するための改善された方法とシステムを提供することである。

【0022】さらに、本発明の目的は、分散データ処理システム内の回復可能なデータベース内部の複数のレプリカ・データベースに付す多様な整合性レベルをユーザが選ぶことができる、改善された方法とシステムを提供することである。

【0023】

【課題を解決するための手段】上述の課題を達成する方法を以下に述べる。分散データ処理システム内の1つまたは複数のロケーションで実施されるジャーナルされたデータベースを含む回復可能なデータベース・システムにおいて、プライマリ・データベースとレプリカ・データベースとの間で維持される整合性レベルをそれぞれ記述する複数の多様な整合性レベルが指定される。ユーザは、各レプリカ・データベースのための具体的な整合性レベルを選択することができる。その後、プライマリ・データベースの中のレコードをアップデートすると、そのアップデートを使って、各レプリカ・データベース内の対応するレコードに対して、そのレプリカ・データベースに選択された整合性レベルと整合のとれる方法でアップデートが始まる。このようにして、プライマリ・データベースが故障した場合にもスムーズに切り替えが行えるように、プライマリ・データベースと完全に整合性のあるレプリカ・データベースを提供することもできるし、また、完全に整合性のあるデータを必要としないアプリケーションの照会に回答するためには第2のレプリカ・データベースを提供し、これにより、そのデータベースに対するアクセスの効率を大きく向上させることもできる。

【0024】

【実施例】図1に、本発明の方法とシステムを実施する

のに使用できる、分散データ処理システム8の概略図を示す。分散データ処理システム8は、ローカルエリア・ネットワーク（以下LANという）10および32のような複数のLANを持ち、各LANはコンピュータ12および30のような複数の個々のコンピュータを持つことが望ましい。もちろん、当業者には明らかなように、各ネットワークには、ホストプロセッサに接続された複数のインテリジェント・ワークステーションを使用することもできる。

【0025】このようなデータ処理システムによく見られるように、各コンピュータは記憶装置14および印刷出力装置16を接続することができる。記憶装置14は、本発明の方法とシステムにしたがい、分散データ処理システム8のユーザによって定期的にアクセスされ処理されるプライマリ・データベースまたはそのレプリカをストアするために使用される。公知の方法によって、そのようなプライマリ・データベースまたはそのレプリカは記憶装置14にストアされ、データベースを維持しアップデートする責任を持つリソース・マネジャーまたはライブラリ・サービスと関連づけられる。

【0026】図1はさらに、分散データ処理システム8が、メインフレーム・コンピュータ18のような複数のメインフレーム・コンピュータを含み、それらが通信リンク22によってLAN10に接続されていることを示す。また、メインフレーム・コンピュータ18は、LAN10の遠隔記憶装置としてはたらく記憶装置20に接続されている。第2のLAN32が、ゲートウェイ・サーバ28への通信制御装置26および通信リンク34を介して、LAN10に接続されている。ゲートウェイ・サーバ28は、LAN32をLAN10につなげる役割をする別個のコンピュータあるいはインテリジェント・ワークステーションであることが望ましい。

【0027】LAN32およびLAN10に関連して述べたように、プライマリ・データベースまたはそのレプリカは記憶装置20内にストアされ、そのようにストアされたプライマリ・データベースおよびそのレプリカに対するリソース・マネジャーまたはライブラリ・サービスの役割をするメインフレーム・コンピュータ18によって制御される。

【0028】もちろん、当業者には明らかなように、メインフレーム・コンピュータ18はLAN10から地理的に遠い距離に設置することができ、同様に、LAN10は、LAN32から相当な距離離れていてもよい。たとえば、LAN32がカリフォルニア州にあり、LAN10がテキサス州にあり、メインフレーム・コンピュータがニューヨークにあってもよい。

【0029】上述したことからわかるように、分散データ処理ネットワーク8の或る1つの部分にいるユーザが、データ処理ネットワーク8の別の部分にストアされているデータベースにアクセスしたいことがしばしばあ

る。そのようなクライアント/サーバ・システム環境では、データベース内にストアされたデータへのアクセスは、いわゆる「回復可能な(resilient)データベース・システム」を提供することによって増やすことができる。回復可能なデータベース・システムとは、分散データ処理システム内の1つまたは複数のロケーションにおいて複製化されている、ジャーナルされたデータベースである。そのようなデータベースでは、1つのデータベースがプライマリ・データベースとして、他の全てのデータベースはバックアップ・レプリカとして指定される。回復可能なデータベース・システムは、故障が起きたとき1つ前のポイントに回復する。すなわち、ここでいう故障とは、データベース・レプリカの1つが在るコンピュータの故障、あるいは、レプリカの1つを破壊するような媒体故障によって引き起こされる故障のことである。クラスター管理を行うことによって、アプリケーション・プログラマおよびエンドユーザに透明で、プライマリ・レプリカが故障したときにバックアップ・レプリカにスムーズに切り替えられる回復処理を提供することができる。

【0030】高い使用可能性を持つデータベースを提供するためにデータベースが複製化されているシステムにおいては、そのようなシステムの自然な延長は、プライマリ・サーバのワークロードを軽減するために、ユーザがバックアップ・レプリカに照会できるようにすることである。しかし、或る整合性が強制されていない限り、レプリカ・データベースに対する照会トランザクションは、コミットしていないトランザクションによってセットされた値、あるいは、古い値を読んでしまう。あまり重要でないアプリケーションではこれでも良いかも知れないが、通常多くのアプリケーションは、プライマリ・データベースとの最低レベルの整合性を必要とする。

【0031】このアプリケーションについて説明する目的で、ここでは、「アプリケーション」という用語は、ユーザによって始動されデータベース・オペレーションを呼び出すアプリケーション・コードのことをいう。データベース・オペレーションを呼び出すために使用される多くの多様なアプリケーションが現在使用可能であり、そのようなアプリケーションは本発明の方法とシステムにとって修正する必要はない。典型的には、そのようなアプリケーションがデータベース・オペレーションを呼び出すときはいつでも使える「アプリケーション・スタブ」も提供される。アプリケーション・スタブは呼び出しをトラップし、その呼び出しを特別ルーチンにまわす。このアプリケーション・スタブは一般的にクラスター管理によって提供され、その後、リクエストを、プライマリ・データベースを制御する機械にわたす。故障を回復するために必要な回復ステップも、アプリケーション・スタブによってアプリケーションから隠されている。

【0032】本発明の明細書で記述される「プライマリ・エージェント」はプライマリ・データベースを制御する機械の上にあり、データベース・オペレーションを実行するプロセスである。アプリケーション・プロセス1つにつき、1つのプライマリ・エージェントがある。本発明の方法とシステムで使用される「バックアップ・レシーバ」は、レプリカ・データベースへのアクセスを管理する機械の上にあり、プライマリ・データベースにどこされた変更を記述するジャーナル・エントリを受け取り、それらのジャーナル・エントリをジャーナルのローカル・レプリカに記録するプロセスである。データベースのレプリカを制御する各機械の上の1つのデータベースに対して1つのバックアップ・レシーバ・プロセスがある。「バックアップ・アプライア」は、レプリカ・データベースを制御する機械の上にあり、データベースのローカル・レプリカにジャーナル・エントリを付すプロセスである。バックアップ・レプリカを制御する各機械の上の1つのデータベースに対して1つのバックアップ・アプライア・プロセスがある。

【0033】図2に、本発明の方法とシステムにしたがい実施された回復可能なデータベース・システム内の、典型的な書き込みオペレーションの制御フローの概略を示す。図に示すように、ユーザ40はアプリケーション42を使用し、スタブ・ルーチン44を呼び出し、プライマリ・データベース46内でオペレーションを実行する。プライマリ・データベース46を制御する機械の中のプライマリ・エージェント48がこのオペレーションを実行する。この修正によって必要になったジャーナル・エントリがジャーナル50内に挿入され、レプリカ・データベース52内のバックアップ・レシーバ54、および、レプリカ・データベース60のような他のレプリカ・データベースに広げられる。

【0034】バックアップ・レシーバ54はそれらのジャーナル・エントリを受け取ったというアクノレジメントを返し、その後、それらのジャーナル・エントリをそのジャーナルのローカルレプリカ（符号58に示すような）内に、随時非同期的に入れる。バックアップ・アプライア56は、ジャーナル58内にあるそれらのジャーナル・エントリをレプリカ・データベース52内の対応するレコードに付ける。これは、そのレコードをロックし、アップデートし、その後、アンロックすることによって行われる。この方法によって、レプリカ・データベース52とプライマリ・データベース46との間の整合性を保つことができる。

【0035】したがって、バックアップ・アプライア56により、ユーザが直接レプリカ・データベース52からレコードを読んだ場合には部分的にしかアップデートされていないレコードを読むのを防ぐことによって、最低レベルの整合性を提供することができる。もちろん、データベース読取りオペレーションに対する制御フロー

は、図2に示したものに似ている。しかし、ジャーナル・エントリは生成されず、したがって、バックアップ・レシーバとバックアップ・アプライアは使用されない。図2に示した環境は、たとえば、IBM AS/400のようなコンピュータ・システムを使用して実施することができる。

【0036】次に、本発明の重要な特徴として、選択されたレプリカ・データベースをユーザが使用する際にユーザが選択する複数の多様な整合性レベルが提供される。回復可能なデータベース・システム内の同時実行については、「同時実行制御とデータベース・システムの回復」(“Concurrency Control and Recovery in Database Systems” Addison-Wesley Publishing Company, 1987)に記載があり、複数バージョンの同時実行制御モデルが記述されている。このシステムにおいては、バックアップ・レプリカ・データベースは常にプライマリ・データベースより遅れ、したがって、データベースのバージョンが2つできる。上記文献のシステムに対比して、本発明の方法とシステムは、考慮すべき2つの異なる整合性属性を提供する。

【0037】具体的には、これらの2つの属性は、データベース・システム内のトランザクションの順序付け(ordering)と、これらのトランザクションの直列化可能性(serializability)を含む。これらの2つの属性はいくつかの方法で組み合わせることができ、その結果6つの異なる整合性レベルができる。したがって、本発明の重要な特徴にしたがい、レプリカ・データベースから読み込む照会に対し、トランザクションの順序付けに関して3つの異なる整合性レベルが考慮される。もちろん、当業者には明かなように、レコード・アップデートの後の整合性維持のために、最大時間遅れの値を指定することができる。「レベル0」整合性レベルは、レプリカ・データベース内の古い値を使って照会することを許す。「レベル1」整合性レベルは、レプリカ・データベースから最新のコミットされた値だけを使って照会できるという整合性を課す。最後に、「レベル2」整合性レベルは、その値がコミットに達したか否かに拘わらず、レプリカ・データベースからの最新の値だけを使って照会するという整合性を課す。

【0038】次に、上述したように、所望の整合性レベルを割り当てるために、本発明の回復可能なデータベース・システム内のトランザクションの直列化可能性について考慮しなければならない。この属性によって、現実には同時にインターリーブして実行されたトランザクションが、或る直列順序で実行されたのと論理的には等しい効果を持たせることを保証する。回復可能なデータベース・システムは、コミットメント制御の下で動くアプリケーションが、コミットメント制御の下にない他のアプリケーションと同時に同じデータベースに対して動くことを許す。したがって、異なるアプリケーションは、

直列化可能性を必要としてもしなくても良いことが考えられる。したがって、本発明の回復可能なデータベース・システム内のレプリカ・データベースにユーザまたはシステム管理者が異なる多様な整合性レベルをセットすることを許すとき、整合性レベルを制御するために2つの異なる変数がセットされなければならない。順序付けは、「レベル0」、「レベル1」、あるいは「レベル2」にセットされる。さらに、直列化可能性は「必要」あるいは「不要」にセットされる。

【0039】図3に、本発明の方法とシステムによる回復可能なデータベース内での複数の多様な整合性レベルのセットを説明する論理フローの概略図を示す。図示したように、このプロセスは、ブロック70で始まり、ブロック72に進む。ブロック72で、整合性レベルの選択が行われたかどうか判断される。Noの場合、このプロセスは、ユーザまたはシステム管理者が回復可能なデータベース・システム内のレプリカ・データベースの整合性レベルをセットしたいという意志表示を示すまで、繰り返される。

【0040】ブロック72で整合性レベルの選択が行われた場合、プロセスはブロック74に進む。ブロック74は、本発明の回復可能なデータベース・システム内の最初のレプリカ・データベースに対するプロセスを示す。次にプロセスはブロック76に進む。ブロック76で、プロセスは、ユーザあるいはシステム・オペレータが所望の整合性レベルをセットすることを促す。次に、プロセスはブロック78に進む。ブロック78で、所望の整合性レベルが入れられたかどうか判断され、Noの場合は、プロセスは繰り返しブロック76に戻り、再度所望の整合性レベルを入れるように促す。

【0041】ブロック78で所望の整合性レベルが入れられたと判断されれば、プロセスはブロック80に進む。ブロック80で、直列化可能性の要求があるかどうかをユーザあるいはシステム・オペレータに促す。すなわち、インターリーブされて実行されたトランザクションが、或る直列順序で実行されたのと論理的に等しいものとしたかどうかを、ユーザあるいはシステム・オペレータに意志表示してもらう。次に、プロセスはブロック82に進む。ブロック82で、ユーザが直列化可能性要求を入れたかどうか判断し、Noの場合は繰り返しブロック80に戻り、再度、このデータベースに対する直列化可能性要求があるか否かを入れるようにユーザに促す。ユーザが直列化可能性要求を入れた後、プロセスはブロック82からブロック84に進む。ブロック84で、整合性レベルを付けるべきレプリカ・データベースがまだあるかどうか判断し、以上のプロセスで整合性レベルを付けたレプリカ・データベースが最後のレプリカでない場合には、プロセスは繰り返しブロック76に戻り、再度上述したプロセスを実行する。ブロック84がYesの場合、すなわち、最後のレプリカ・データベ

ースに所望の整合性レベルが付けられた場合は、プロセスはブロック86に進みリターンする。

【0042】図4から図10に、本発明の方法とシステムによる回復可能なデータベース・システム内で、レコード・アップデートが行われた後の、複数の多様な整合性レベルを実行する方法を示すフローチャートの概略図を示す。図4に示すように、プロセスはブロック90から始まる。次に、プロセスはブロック92に進み、レコード・アップデートが起こったかどうか判断する。レコード・アップデートがまだ起こっていない場合、このプロセスは、プライマリ・データベースにレコード・アップデートが起こるまで繰り返される。次に、レコード・アップデートが起こったときに、プロセスはブロック92からブロック94に進む。

【0043】ブロック94は、本発明の回復可能なデータベース・システム内の各レプリカ・データベースに対して繰り返して行われるプロセスの始まりを示す。プロセスはブロック94からブロック96に進む。ブロック96で、所望の整合性レベルが「レベル0」にセットされたかどうか判断される。Yesであれば、プロセスはブロック98に進む。ブロック98で直列化可能性が要求されているかどうか判断され、Yesの場合、プロセスは、図4のコネクタ100を介して図5の論理フローに進む。そうでない場合、すなわち、直列化可能性が不要であった場合、プロセスは、図4のコネクタ102を介して図6の論理フローに進む。

【0044】再びブロック96に戻り、所望の整合性レベルが「レベル0」にセットされなかった場合は、プロセスはブロック104に進む。ブロック104で、所望の整合性レベルが「レベル1」にセットされたかどうか判断される。Yesであれば、プロセスはブロック104からブロック106に進み、再度、直列化可能性が要求されているかどうか判断される。直列化可能性が要求されている場合には、プロセスは、図4のコネクタ108を介して図7の論理フローに進む。そうでない場合、すなわち、直列化可能性が不要であった場合、コネクタ110を介して図8の論理フローに進む。

【0045】再びブロック104に戻り、所望の整合性レベルが「レベル1」でない場合、プロセスはブロック104からブロック112に進む。ブロック112で、このレプリカ・データベースに対する所望の整合性レベルが「レベル2」にセットされたかどうか判断され、もしYesであれば、プロセスはブロック112からブロック114に進む。前と同じように、ブロック114で直列化可能性が要求されたかどうか判断され、Yesの場合、プロセスはコネクタ116を介して図9の論理フローに進む。Noの場合、すなわち、直列化可能性が不要であった場合、プロセスはブロック114からコネクタ118を介して図10の論理フローに進む。最後に、再びブロック112で、所望の整合性レベルが「レ

ベル2」でない場合、プロセスはブロック120に進みリターンする。

【0046】図5に、レプリカ・データベース内で整合性「レベル0」が選択され直列化可能性が要求されたレコードに対応するレコードにレコード・アップデートが行われた場合に起こる論理フローの順序を示す。図5から図10のそれぞれには、番号130、132、134、および、136を付けた4つのカラムがある。それぞれの場合において、カラム130はプライマリ・データベースに起こるアクティビティを示す。カラム132はジャーナル・エントリのフローを示し、カラム134は考慮対象の特定のレプリカ・データベース内のバックアップ・レシーバに起こるアクティビティを示す。最後に、カラム136は、レプリカ・データベース内のバックアップ・アプライア（図2を参照）のアクティビティを示す。

【0047】したがって、図5のカラム130を見ると、ブロック150に、レコードR1に対するアップデートU1を含むトランザクションT1が示されている。この結果ジャーナル・エントリ（JE1）が作られ、これがバックアップ・レシーバに送られる。次に、バックアップ・レシーバはアクノレジメンをプライマリ・データベースに送り、ジャーナル・エントリ（JE1）を、レプリカ・データベースにあるローカルジャーナルに入れる。次に、バックアップ・アプライアはレコードR1をロックし、ジャーナル・エントリ（JE1）を付ける。

【0048】次に、ブロック152に、レコードR2に対するアップデートU2を含むトランザクションT2が示されている。この結果ジャーナル・エントリ（JE2）が作られ、これがバックアップ・レシーバに送られる。次に、バックアップ・レシーバはアクノレジメンをプライマリ・データベースに送り、ジャーナル・エントリ（JE2）をローカル・ジャーナルに入れる。その後、バックアップ・アプライアはレコードR2をロックし、ジャーナル・エントリ（JE2）を付ける。

【0049】次に、ブロック154に、レコードR3に対するアップデートU3を含むトランザクションT1を示す。このトランザクションの結果第3のジャーナル・エントリ（JE3）が作られ、これがバックアップ・レシーバに送られる。バックアップ・レシーバは再度アクノレジメンを送り、このジャーナル・エントリをローカルジャーナルに入れる。次に、バックアップ・アプライアはレコードR3をロックし、ジャーナル・エントリ（JE3）を付ける。

【0050】最後に、ブロック156に、トランザクションT1のためのコミット・オペレーションを示す。このトランザクションの結果、ジャーナル・エントリ（JE4）が作られ、バックアップ・レシーバに送られる。バックアップ・レシーバはアクノレジメンをプライ

マリ・データベースに送り、ジャーナル・エントリ（JE4）をローカルジャーナルに入れる。次に、バックアップ・アプライアはレコードR1とR3に対するロックをリリースする。

【0051】図6は、整合性レベルを「レベル0」にセットし直列化可能性を不要にしたレプリカ・データベース内のレコードにアップデートをした結果起こる論理フローを示す。説明を簡単にするために、図6から図10のカラムおよびトランザクションは、図5に使ったものと同じ番号を使用している。上述したのと同じように、プロセスは、レコードR1に対するアップデートU1を含むトランザクションT1を示すブロック150から始まる。このトランザクションの結果、ジャーナル・エントリ（JE1）が作られ、これがバックアップ・レシーバに結合される。アクノレジメンがバックアップ・レシーバからプライマリ・データベースに送られ、ジャーナル・エントリ（JE1）がローカル・ジャーナルに入れられる。次に、バックアップ・アプライアはレコードR1をロックし、ジャーナル・エントリ（JE1）を付ける。その後、直列化可能性が不要なので、バックアップ・アプライアはレコードR1をリリースする。

【0052】ブロック152に、レコードR2に対するアップデートU2を含むトランザクションT2を示す。このトランザクションの結果ジャーナル・エントリ（JE2）が作られ、これがバックアップ・レシーバに送られる。バックアップ・レシーバはアクノレジメンをプライマリ・データベースに送り、ジャーナル・エントリ（JE2）をローカルジャーナルに入れる。次に、バックアップ・アプライアはレコードR2をロックし、ジャーナル・エントリ（JE2）を付け、その後、レコードR2をリリースする。

【0053】ブロック154に、レコードR3に対するアップデートU3を含むトランザクションT1の追加部分を示す。このトランザクションの結果ジャーナル・エントリ（JE3）が作られ、バックアップ・レシーバに送られる。バックアップ・レシーバはアクノレジメンをプライマリ・データベースに送り、ジャーナル・エントリ（JE3）をローカルジャーナルに入れる。次に、バックアップ・アプライアはレコードR3をロックし、ジャーナル・エントリ（JE3）を付け、その後、レコードR3をリリースする。

【0054】最後に、ブロック156に、トランザクションT1に対するコミット・オペレーションを示す。このコミット・オペレーションの結果、ジャーナル・エントリ（JE4）が作られ、バックアップ・レシーバに送られる。アクノレジメンがバックアップ・レシーバによりプライマリ・データベース送られ、ジャーナル・エントリ（JE4）がローカルジャーナルに入れられる。直列化可能性が不要であるので、このコミット・オペレーションの結果、バックアップ・アプライアによる

17

アクティビティは起こらない。

【0055】次に、図7に、レプリカ・データベースに対して整合性「レベル1」が選択され直列化可能性が要求されている場合のプライマリ・データベース内のレコードに対するアップデートへの応答の方法を示す論理フローを示す。前と同じように、ブロック150は、レコードR1に対するアップデートU1を含むトランザクションT1を示す。このトランザクションの結果ジャーナル・エントリ（JE1）が作られ、バックアップ・レシーバに送られる。アクノレジメントがバックアップ・レシーバによってプライマリ・データベースに送られ、ジャーナル・エントリ（JE1）がローカルジャーナルに入れられる。次に、バックアップ・アプライアはこのジャーナル・エントリ（JE1）をバッファにストアする。上述したように、整合性「レベル1」は、照会がコミットされた最新の値のみを読むことを課しているの
 で、当業者には明かなように、レコードR1に対するアップデートは、コミット・トランザクションが起こるまでの間、必ずバッファにストアされなければならない。

【0056】次に、ブロック152に示すように、レコードR2に対するアップデートU2を含むトランザクションT2が起こる。この結果ジャーナル・エントリ（JE2）が作られ、バックアップ・レシーバに結合される。バックアップ・レシーバは、アクノレジメントをプライマリ・データベースに送り、ジャーナル・エントリ（JE2）をローカルジャーナルに入れる。前と同じように、次に、バックアップ・アプライアはこのジャーナル・エントリ（JE2）をバッファにストアする。

【0057】次に、ブロック154に示すように、レコードR3に対するアップデートU3を含むトランザクションT1が起こる。この結果ジャーナル・エントリ（JE3）が作られ、バックアップ・レシーバに結合される。前と同じように、アクノレジメントがバックアップ・レシーバからプライマリ・データベースに送られ、ジャーナル・エントリ（JE3）がローカルジャーナルに入れられる。バックアップ・アプライアはこのジャーナル・エントリ（JE3）をバッファ内にストアする。

【0058】最後に、ブロック156に示すように、トランザクションT1のためのコミット・オペレーションが起こる。この結果、ジャーナル・エントリ（JE4）が作られ、バックアップ・レシーバに送られる。バックアップ・レシーバはレコードR1とR3をロックし、アクノレジメントをプライマリ・データベースに送る。その後、バックアップ・アプライアはトランザクションT1からジャーナル・エントリを取り出し、該当するジャーナル・エントリ、すなわち、トランザクションT1に関連しているジャーナル・エントリ（JE1およびJE3）を付ける。その後、レコードR1およびR3はリリースされる。したがって、レコードR1およびR3に

18

対するアップデートは、そのトランザクションのためのコミット・オペレーションが起こるまで、バックアップ・アプライアによって適用されない。

【0059】図8に、整合性レベルを「レベル1」に、直列化可能性を不要にセットしたレプリカ・データベース内でアップデートされたレコードに対する応答方法を記述した論理フローを示す。前と同じように、ブロック150は、レコードR1に対するアップデートU1を含むトランザクションT1を示す。この結果、ジャーナル・エントリ（JE1）が作られ、バックアップ・レシーバに結合される。バックアップ・レシーバはアクノレジメントをプライマリ・データベースに送り、このジャーナル・エントリ（JE1）をローカルジャーナルに入れる。次に、ジャーナル・エントリ（JE1）は、バックアップ・アプライアによってバッファにストアされる。

【0060】次に、ブロック152に示すように、レコードR2に対するアップデートU2を含むトランザクションT2が起こる。この結果、ジャーナル・エントリ（JE2）が作られ、バックアップ・レシーバに結合される。次に、バックアップ・レシーバはアクノレジメントをプライマリ・データベースに送り、このジャーナル・エントリ（JE2）をローカルジャーナルに入れる。次に、前と同じように、このジャーナル・エントリはバックアップ・アプライアによってバッファにストアされる。

【0061】ブロック154で、レコードR3に対するアップデートU3を含むトランザクションT1が起こる。この結果ジャーナル・エントリ（JE3）が作られ、バックアップ・レシーバに結合される。アクノレジメントがバックアップ・レシーバからプライマリ・データベースに送られ、ジャーナル・エントリ（JE3）がローカルジャーナル内にストアされる。次に、前と同じように、ジャーナル・エントリ（JE3）がバックアップ・アプライアによってバッファにストアされる。

【0062】最後に、ブロック156に示すように、トランザクションT1のためのコミット・オペレーションが起こる。この結果ジャーナル・エントリ（JE4）が作られ、バックアップ・レシーバに結合される。次に、バックアップ・レシーバはレコードR1とR3をロックし、アクノレジメントをプライマリ・データベースに送る。その後、バックアップ・アプライアはトランザクションT1に対するジャーナル・エントリを取り出し、第1のジャーナル・エントリ（JE1）を適用する。次に、レコードR1がリリースされる。次に、トランザクションT1に対する第2のジャーナル・エントリ（JE3）がレコードR3に適用され、そのレコードはリリースされる。図7の論理フローとは対照的に、図8の論理フローでは直列化可能性の要求がないので、各レコードはそのアップデートが適用された後でリリースされる点

に留意する必要がある。

【0063】次に、図9に、整合性レベルを「レベル2」にセットし直列化可能性を必要とするプライマリ・データベース内の対応するレコードのアップデートに対する、レプリカ・データベースの応答方法を記述した論理フローを示す。前と同じように、ブロック150で、レコードR1に対するアップデートU1を含むトランザクションT1が起こる。このトランザクションの結果、ジャーナル・エントリ（JE1）が作られ、バックアップ・レシーバに送られる。次に、バックアップ・レシーバはR1をロックし、アクノレジメントをプライマリ・データベースに送る。次に、ジャーナル・エントリ（JE1）がローカルジャーナルに入れられ、バックアップ・アプライアは直ちにジャーナル・エントリ（JE1）を適用する。当業者には明かなように、整合性「レベル2」は、照会は最新の値（コミットが起こったかどうかにかかわらず）のみを読むことによって行うことを課している。したがって、レコードR1に対するアップデートは、バックアップ・アプライアによって直ちに適用される。

【0064】ブロック152で、レコードR2に対するアップデートU2を含むトランザクションT2が起こる。この結果、ジャーナル・エントリ（JE2）が作られ、バックアップ・レシーバに結合される。次に、バックアップ・レシーバはレコードR2をロックし、アクノレジメントをプライマリ・データベースに送り、ジャーナル・エントリ（JE2）をローカルジャーナルに入れる。次に、バックアップ・アプライアは直ちにジャーナル・エントリ（JE2）を適用し、レコードR2を最新の値にアップデートする。

【0065】ブロック154で、レコードR3に対するアップデートU3を含むトランザクションT1の第2部分が起る。この結果、ジャーナル・エントリ（JE3）が作られてバックアップ・レシーバに結合される。次に、バックアップ・レシーバはレコードR3をロックし、アクノレジメントをプライマリ・データベースに送る。次に、ジャーナル・エントリ（JE3）がローカルジャーナルに入れられ、バックアップ・アプライアはこのジャーナル・エントリ（JE3）をレコードR3に適用する。

【0066】最後に、ブロック156で、トランザクションT1に対するコミット・オペレーションが起こる。この結果、ジャーナル・エントリ（JE4）が作られ、バックアップ・レシーバに結合される。次に、バックアップ・レシーバはアクノレジメントをプライマリ・データベースに送り、このジャーナル・エントリ（JE4）をローカルジャーナルに入れる。次に、バックアップ・アプライアはレコードR1とR3をリリースし、必要に応じて、このトランザクション部分の直列化可能性を維持する。

【0067】最後に、図10に、整合性「レベル2」をセットし直列化可能性を要求しないレプリカ・データベースの中のプライマリ・データベース内のレコードに対するアップデートに回答して起こる論理フローを示す。前と同様に、ブロック150で、レコードR1に対するアップデートU1を含むトランザクションT1が起こる。この結果、ジャーナル・エントリ（JE1）が作られバックアップ・レシーバに結合される。次に、バックアップ・レシーバはレコードR1をロックし、アクノレジメントをプライマリ・データベースに送る。次に、ジャーナル・エントリ（JE1）がローカルジャーナルに入れられる。次に、バックアップ・アプライアはジャーナル・エントリ（JE1）を適用し、直列化可能性が要求されていないので、直ちにレコードR1をリリースする。

【0068】次に、ブロック152に示すように、レコードR2に対するアップデートU2を含むトランザクションT2が起こる。この結果、ジャーナル・エントリ（JE2）が作られバックアップ・レシーバに結合される。次に、バックアップ・レシーバはレコードR2をロックし、アクノレジメントをプライマリ・データベースに送る。次に、ジャーナル・エントリ（JE2）がローカルジャーナルに入れられる。次に、バックアップ・アプライアはジャーナル・エントリ（JE2）を適用し、レコードR2をリリースする。

【0069】次に、ブロック154に示すように、レコードR3に対するアップデートU3を含むトランザクションT1の次の部分が起る。この結果、ジャーナル・エントリ（JE3）が作られ、バックアップ・レシーバに結合される。次に、バックアップ・レシーバはレコードR3をロックし、アクノレジメントをプライマリ・データベースに送る。次に、ジャーナル・エントリ（JE3）がローカルジャーナルに入れられ、バックアップ・アプライアはジャーナル・エントリ（JE3）を適用し、レコードR3をリリースする。以上の記述から当業者には明かなように、この整合性レベルでは直列化可能性が要求されていないので、レコードR1およびR3に対するアップデートが行われ、各レコードは、そのアップデートが行われた後直ちにリリースされる。

【0070】最後に、ブロック156で、トランザクションT1に対するコミット・オペレーションが起こる。この結果、ジャーナル・エントリ（JE4）が作られ、バックアップ・レシーバに結合される。次に、バックアップ・レシーバはこのジャーナル・エントリのアクノレジメントを送り、ジャーナル・エントリ（JE4）をローカルジャーナルに入れる。

【0071】まとめとして、本発明の構成に関して以下の事項を開示する。

（1）分散データ処理システム内でのデータベース・アクセス効率を向上させる方法であって、前記分散データ

10

20

30

40

50

処理システム内において、ストアされた複数のレコードを持つ1つのデータベースを選択しプライマリ・データベースとして指定するステップと、前記分散データ処理システム内の第2の物理的ロケーションにおいて前記プライマリ・データベースを複製するステップと、前記プライマリ・データベースと前記の複製されたレプリカ・データベースとの間に維持される複数の多様な整合性レベルを指定するステップと、ユーザに前記の複数の多様な整合性レベルから1つを指定させるステップと、を有し、前記プライマリ・データベース内のレコードに対するアップデートにตอบสนองして、前記プライマリ・データベースと前記レプリカ・データベースとの間に前記の複数の多様な整合性レベルの前記の指定された1つを自動的に維持する前記方法。

(2) 前記分散データ処理システム内の第3の物理的ロケーションにおいて前記プライマリ・データベースを複製するステップをさらに有する、前記(1)に記載の方法。

(3) ユーザに前記の複数の多様な整合性レベルから1つを指定させる前記ステップが、前記プライマリ・データベースとそのそれぞれのレプリカとの間に維持される前記の複数の多様な整合性レベルから別の1つを指定させるステップを有する、前記(2)に記載の方法。

(4) 分散データ処理システム内でのデータベース・アクセス効率を向上させるシステムであって、前記分散データ処理システム内において、ストアされた複数のレコードを持つプライマリ・データベースと、前記分散データ処理システム内の第2の物理的ロケーションにおいてストアされている前記プライマリ・データベースのレプリカと、前記プライマリ・データベースと前記レプリカ・データベースとの間に維持される複数の多様な整合性レベルを指定する手段と、ユーザに前記の複数の多様な整合性レベルから1つを指定させる手段と、前記プライマリ・データベース内のレコードに対するアップデートにตอบสนองして、前記プライマリ・データベースと前記レプリカ・データベースとの間に前記の複数の多様な整合性レベルの前記の指定された1つを自動的に維持する手段と、を有するシステム。

(5) 前記分散データ処理システム内の第3の物理的ロケーションにおいてストアされている前記プライマリ・データベースの第2のレプリカをさらに有する、前記(4)に記載のシステム。

(6) ユーザに前記の複数の多様な整合性レベルから1つを指定させる前記手段が、前記プライマリ・データベースとそのそれぞれのレプリカとの間に維持される前記の複数の多様な整合性レベルから別の1つを指定させる手段を有する、前記(5)に記載のシステム。

(7) 分散データ処理システム内でのデータベース・アクセス効率を向上させる方法であって、前記分散データ処理システム内において、ストアされた複数のレコード

を持つ1つのデータベースを選択しプライマリ・データベースとして指定するステップと、前記分散データ処理システム内の第2の物理的ロケーションにおいて前記プライマリ・データベースを複製するステップと、前記プライマリ・データベース内の1つの選択されたレコードに対するアップデートにตอบสนองして、前記の複製されたデータベース内の対応する選択されたレコードへのアクセスを自動的にロックするステップと、前記の複製されたデータベース内の前記の選択されたレコードをアップデートするステップと、前記アップデートの後で、前記の選択されたレコードへのアクセスをリリースするステップと、前記分散データ処理システム内のユーザに、前記プライマリ・データベースおよび前記の複製されたデータベースの中のレコードを照会させるステップと、を有し、データの絶対的な整合性を維持するとともにデータアクセスの使用可能性を向上させる方法。

(8) 前記プライマリ・データベース内のレコードに対する各アップデートが、コミット・オペレーションを行う前の1つの整合性時点に戻り、前記の選択されたレコードへのアクセスを前記アップデートの後でリリースする前記ステップが、前記アップデートに続くコミット・オペレーションの後でのみ前記の選択されたレコードへのアクセスをリリースするステップを有する、前記

(7)に記載の方法。

(9) 分散データ処理システム内でのデータベース・アクセス効率を向上させるシステムであって、前記分散データ処理システム内において、ストアされた複数のレコードを持つプライマリ・データベースと、前記分散データ処理システム内の第2の物理的ロケーションにおいてストアされている前記プライマリ・データベースのレプリカと、前記プライマリ・データベース内の1つの選択されたレコードに対するアップデートにตอบสนองして、前記の複製されたデータベース内の対応する選択されたレコードへのアクセスを自動的にロックする手段と、前記の複製されたデータベース内の前記の選択されたレコードをアップデートする手段と、前記アップデートの後で、前記の選択されたレコードへのアクセスをリリースする手段と、前記分散データ処理システム内のユーザに、前記プライマリ・データベースおよび前記の複製されたデータベースの中のレコードを照会させる手段と、を有し、データの絶対的な整合性を維持するとともにデータアクセスの使用可能性を向上させるシステム。

(10) 前記プライマリ・データベースを、いかなる時点でも、コミット・オペレーションを行う前の整合性時点に戻す手段をさらに有する、前記(9)に記載のシステム。

(11) 前記アップデートの後で前記の選択されたレコードへのアクセスをリリースする前記手段が、前記アップデートに続くコミット・オペレーションの後でのみ前記の選択されたレコードへのアクセスをリリースする手

段を有する、前記(10)に記載のシステム。

【0072】

【発明の効果】本発明の方法とシステムによって、ユーザあるいはシステム・オペレータは、バックアップ・サーバ内のレプリカ・データベースに適する任意の整合性レベルを広範な整合性レベルから選択できるようになる(この選択には、整合性と性能との間のトレードオフが含まれる)。この方法とシステムにより、最低限の整合性レベルでは、1つのレプリカ・データベースにおいて、古い、コミットされていない値を読むことができる一方、他方では、プライマリ・データベースの第2のレプリカ内で、あたかも、照会がプライマリ・データベースから読み込みを行ったのと同じ結果を返すことができる最大レベルの整合性を提供することもできる。このようにして、上述した回復可能なデータベース・システム内において、データベースへのアクセスの効率を大いに改善することができる。

【図面の簡単な説明】

【図1】本発明の方法およびシステムを実施するのに使用できる分散データ処理システムの概略図である。

【図2】本発明の方法およびシステムにしたがって実施される回復可能なデータベース・システム内での、典型的な書き込みオペレーションの概略制御フローを示す。

【図3】本発明の方法およびシステムによる回復可能なデータベース内での複数の多様な整合性レベルの設定を図示する概略論理フローチャートである。

【図4】本発明による複数の多様な整合性レベルの実施を図示する概略論理フローチャートであり、図5に続

く。

【図5】図4に続く論理フローチャート。

【図6】図5に続く論理フローチャート。

【図7】図6に続く論理フローチャート。

【図8】図7に続く論理フローチャート。

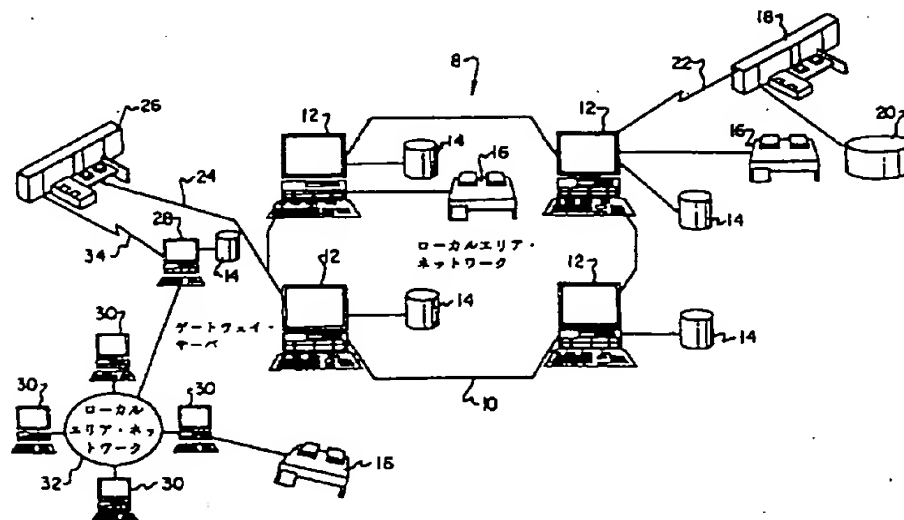
【図9】図8に続く論理フローチャート。

【図10】図9に続く論理フローチャート。

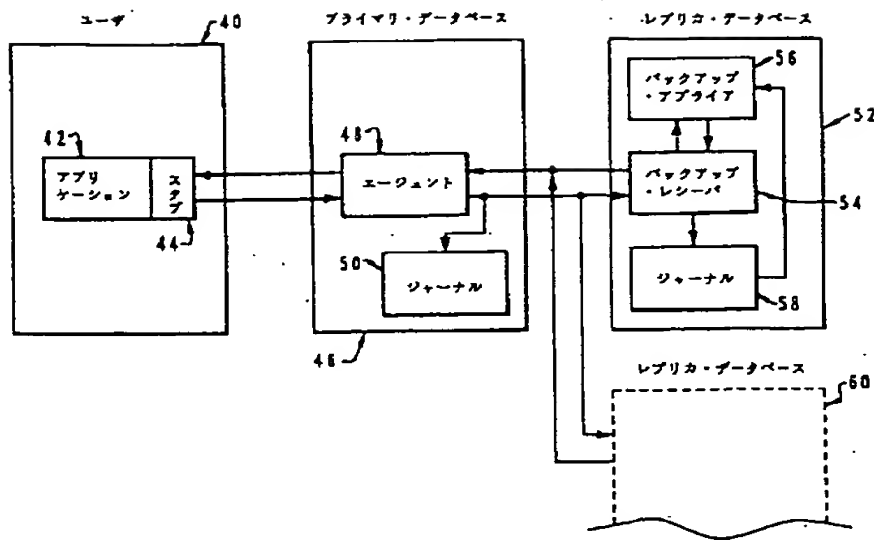
【符号の説明】

8	分散データ処理システム
10、32	ローカルエリア・ネットワーク (LAN)
12、30	コンピュータ
14、20	記憶装置
16	印刷出力装置
18	メインフレーム・コンピュータ
22、24、34	通信リンク
26	通信制御装置
28	ゲートウェイ・サーバ
40	ユーザ
42	アプリケーション
44	スタブ・ルーチン
46	プライマリ・データベース
48	プライマリ・エージェント
50、58	ジャーナル
52	レプリカ・データベース
54、60	バックアップ・レシーバ
56	バックアップ・アプライア

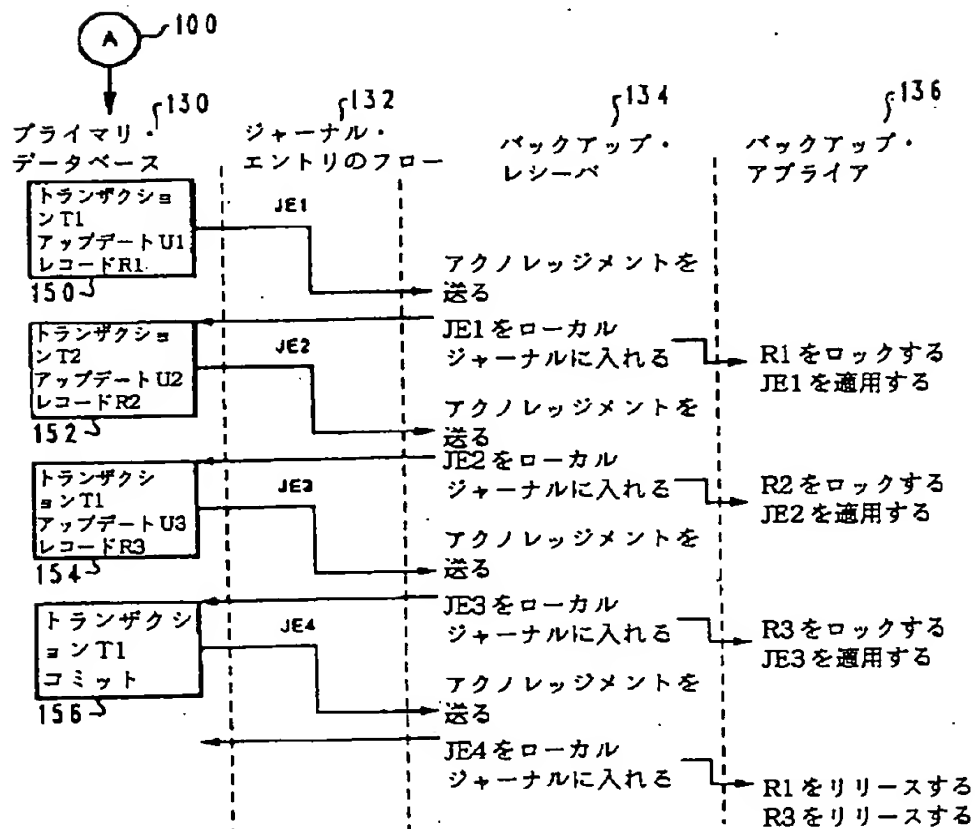
【図1】



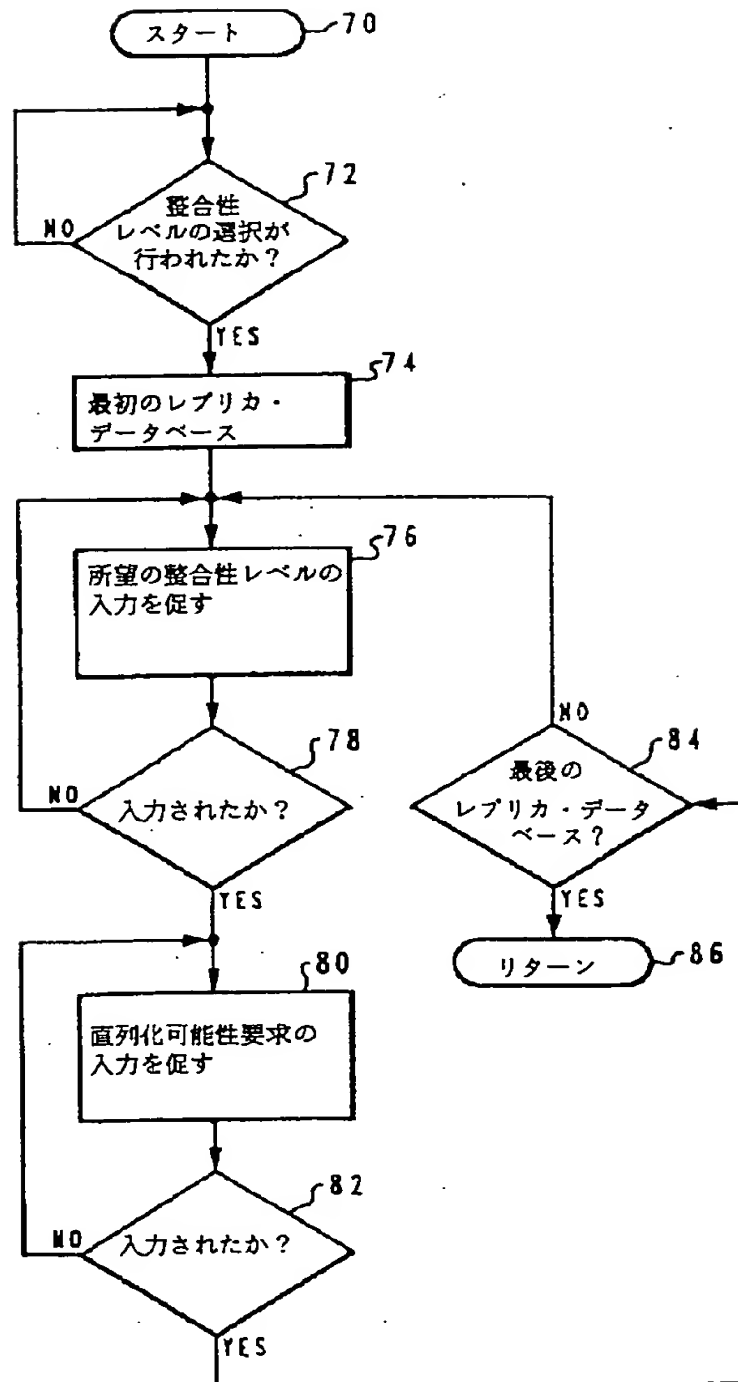
【図2】



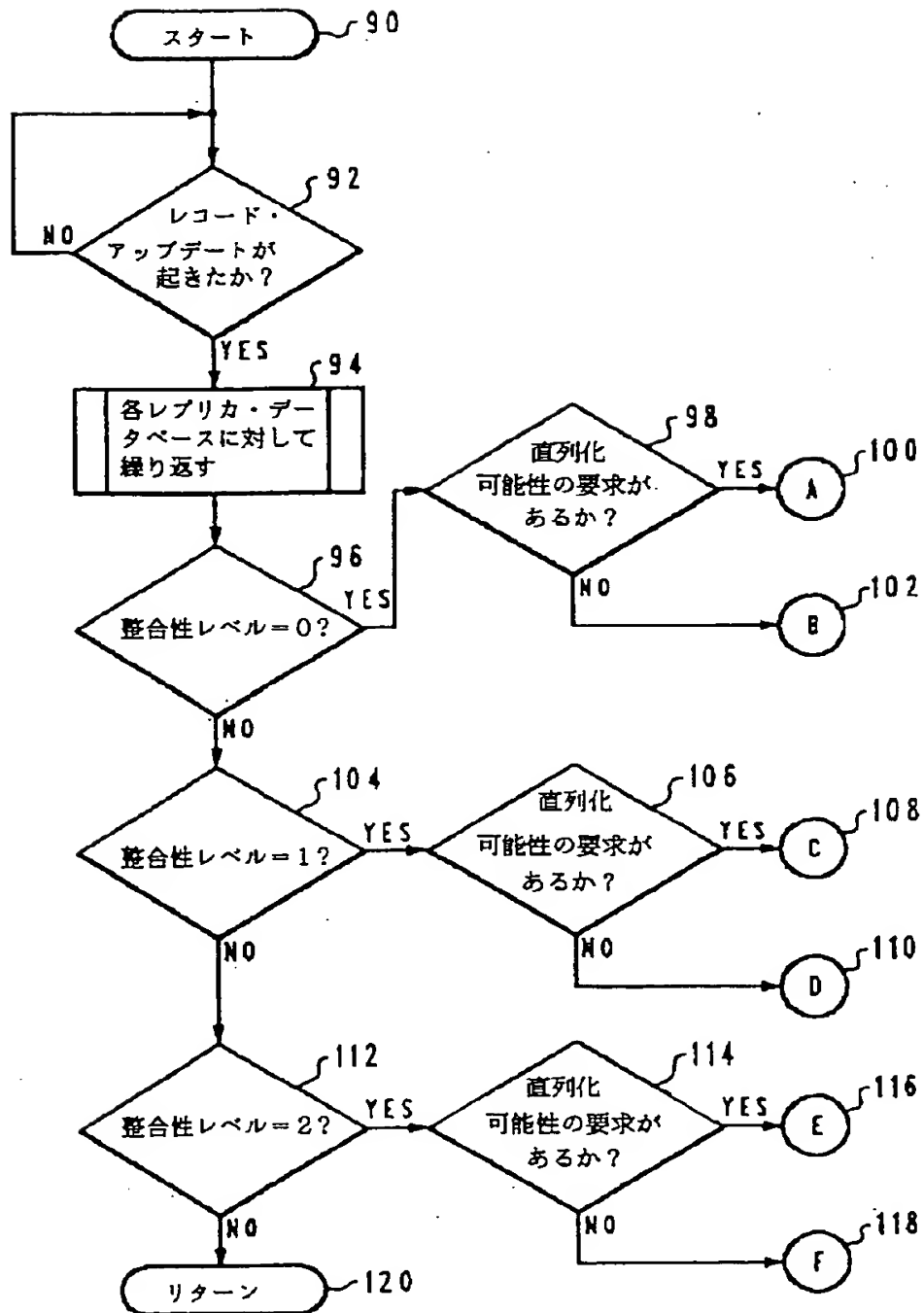
【図5】



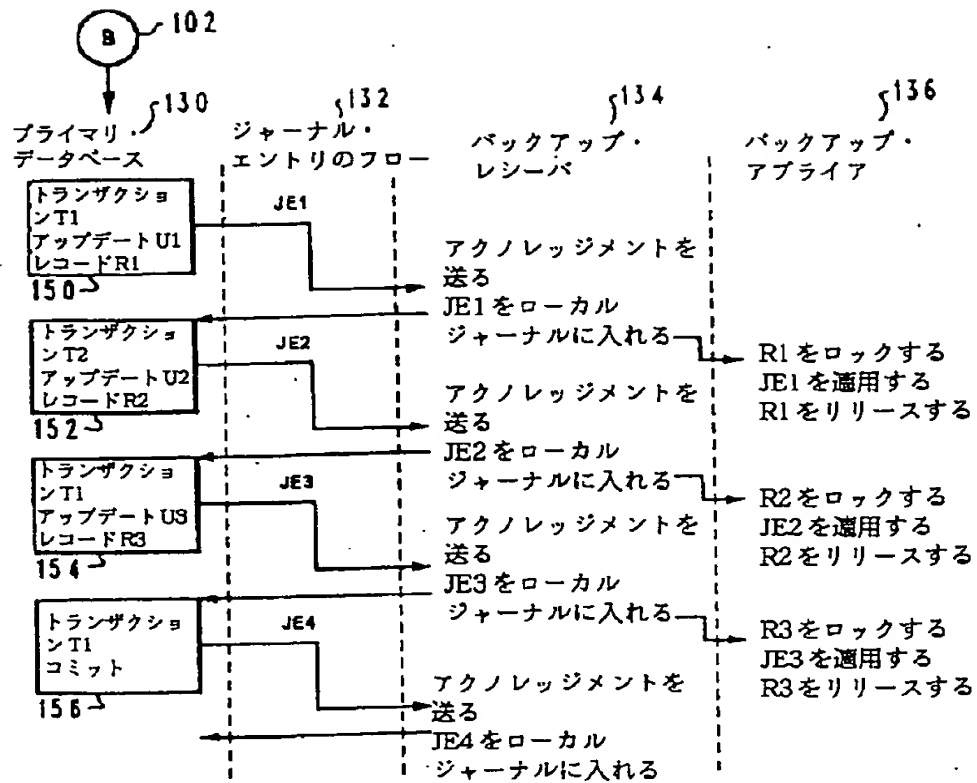
【図 3】



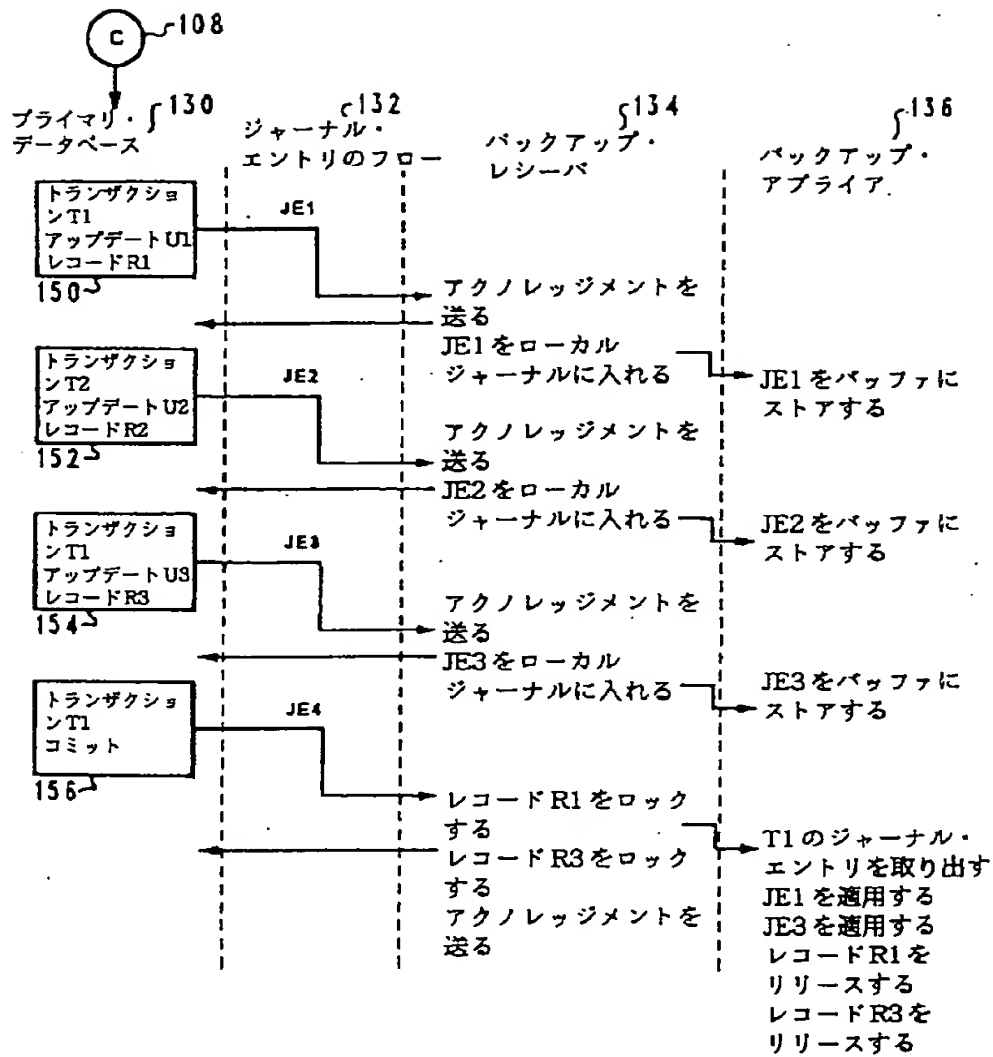
【図 4】



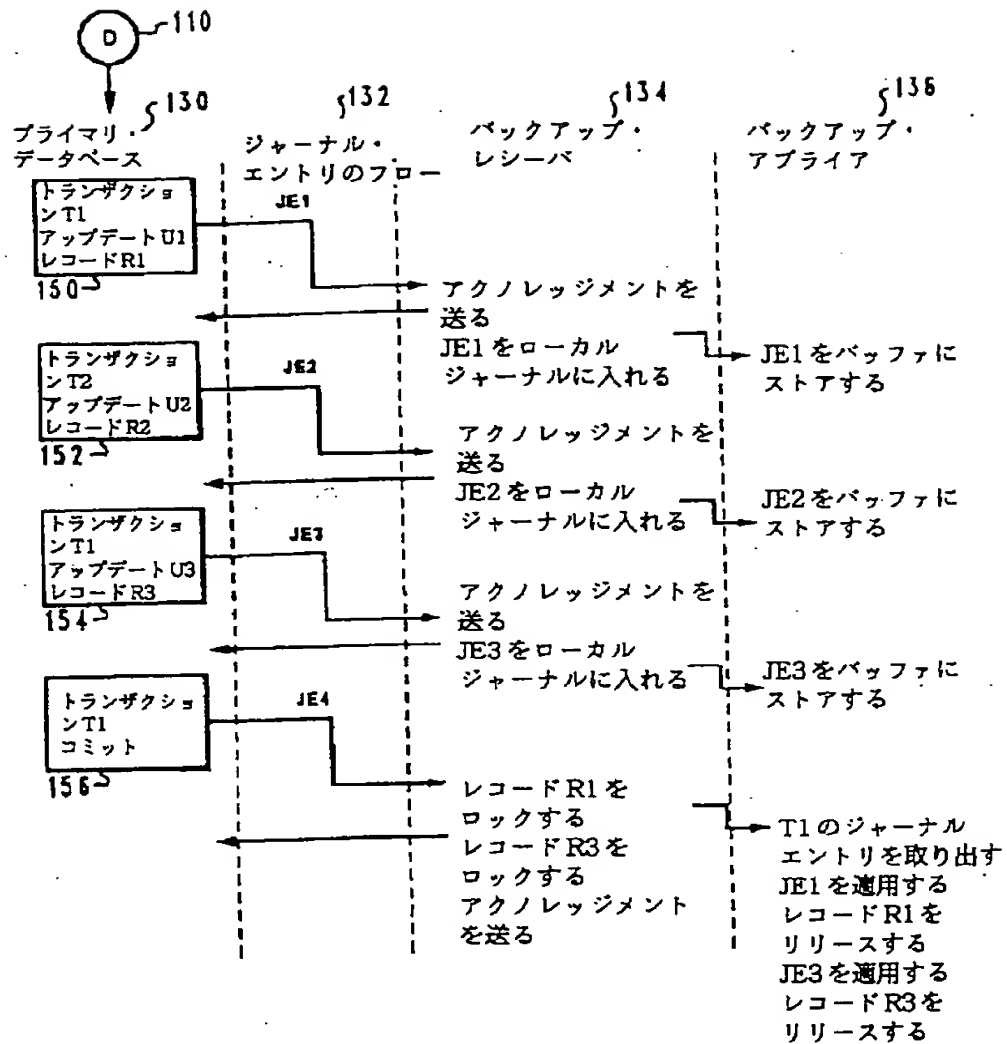
【図6】



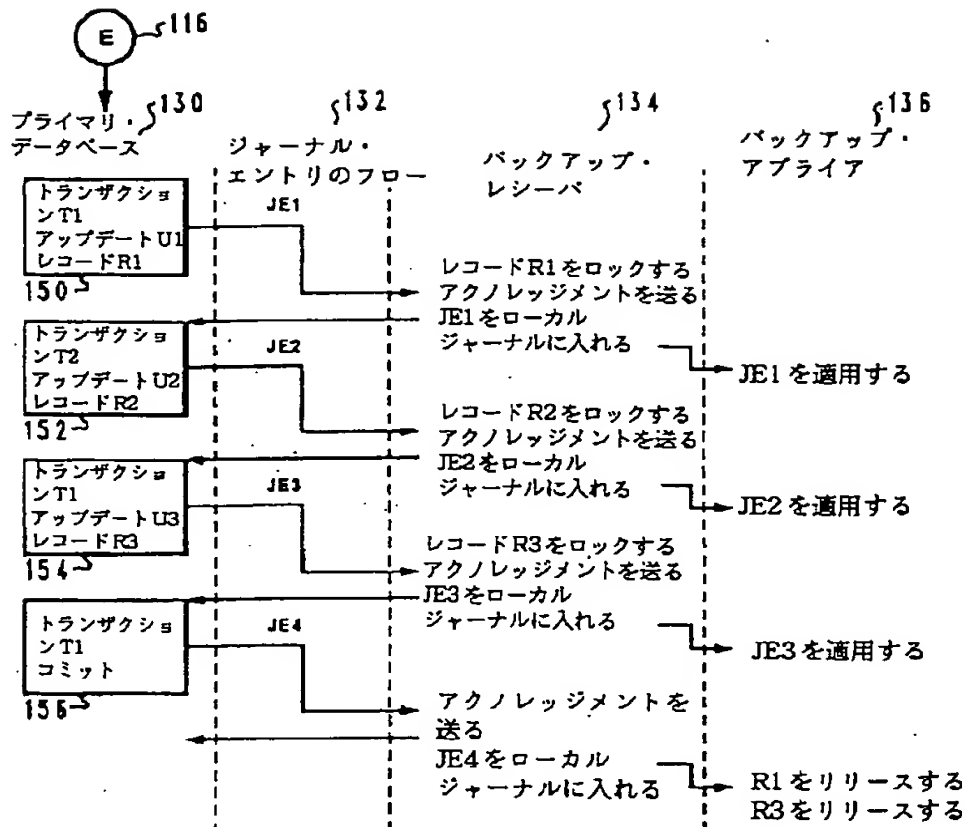
【図 7】



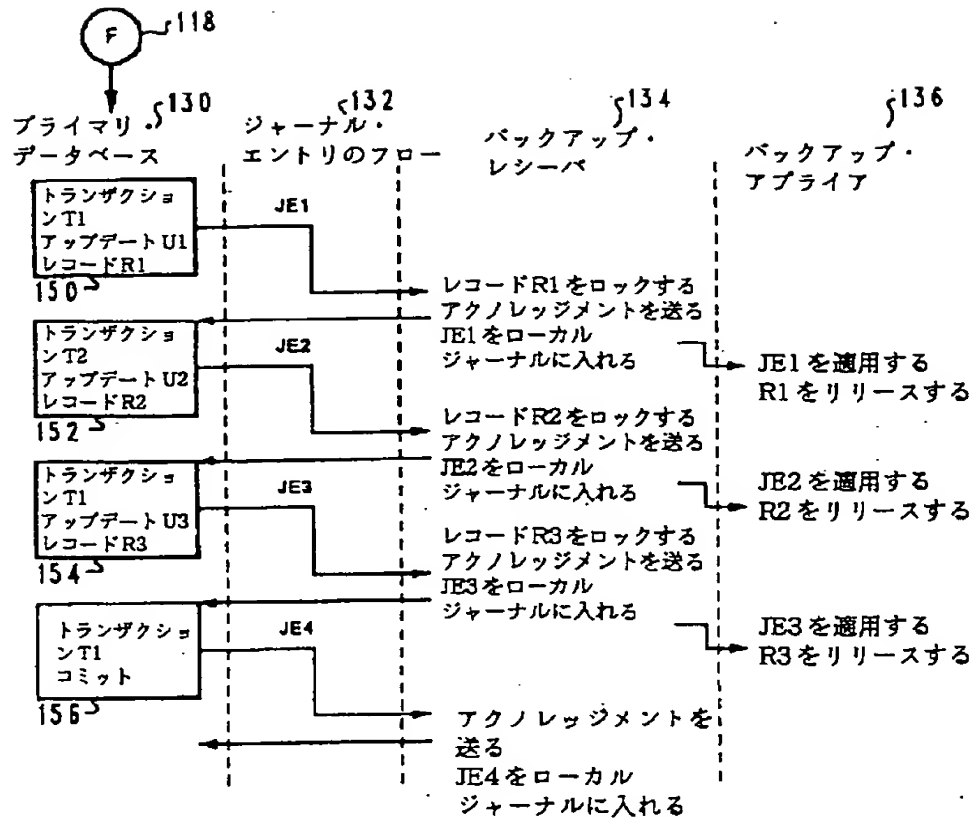
【図8】



【図9】



【図10】



フロントページの続き

(72)発明者 ダニー・ドレブ
イスラエル、メバセレット・イエルチャラ
イム、メボ・ナアマ 28

(72)発明者 ジャーマン・ゴフト
アメリカ合衆国13760ニューヨーク州エン
ディコット、ボーク・ヒル・ロード
207 アパートメント 23

(72)発明者 ジョン・エム・マーバーグ
イスラエル34754ハイファ、ピトキン・ス
トリート 31

(72)発明者 ジェイムス・ジー・ランワイラー
アメリカ合衆国55902ミネソタ州ロチェス
ター、グレンクロフト・レイン・サウスウ
ェスト 5720

(72)発明者 ジュリアン・サトラン
アメリカ合衆国10589ニューヨーク州ソマ
ーズ、ヘリテッジ・ヒルズ 82エイ